Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur

Practice Sheet #05

Topic: Functions in C

Date: 30-01-2017

Instructions:
For the questions consisting code segments, assume necessary libraries have been included. In case, if you feel any discrepancy, kindly state relevant assumption with proper reason.

Every question has only one correct option among given four options. If you feel none of the option is correct then state your answer with proper reason.

1. Consider the following C function, for large values of y, the return value of the function f best approximates

```
float f(float x, int y)
{
  float p, s; int i;
  for (s=1, p=1, i=1; i < y; i ++)
  {
    p*= x/i;
    s+=p;
  }
  return s;
}
```

(a) x^y
(b) e^x
(c) ln(1+x)
(d) x^x

2. Consider the following C-program:

```
void foo(int n, int sum)
{
  int k = 0, j = 0;
  if (n == 0) return;
    k = n % 10;
  j = n / 10;
  sum = sum + k;
  foo (j, sum);
  printf ("%d,", k);
}
int main (){
   int a = 2048, sum = 0;
   foo (a, sum);
   printf ("%d\n", sum);
   getchar();
}
```

What does the above program print?

(a) 8, 4, 0, 2, 14
(b) 8, 4, 0, 2, 0
(c) 2, 0, 4, 8, 14
(d) 2, 0, 4, 8, 0

3. Consider the following C-program:

```
double foo (double); /* Line 1 */
    int main()
   {
     double da, db;
     db = foo(da);
   }
   double foo(double a)
  {
     return a;
  }
```

The above code compiled without any error or warning. If Line 1 is deleted, the above code will show:

(a) no compile warning or error
(b) some compiler-warnings not leading to unintended results
(c) some compiler-warnings due to type-mismatch eventually leading to unintended results
(d) compile error

4. Figure out the output of the below code. State relevant assumption if any during computation. Assume Necessary libraries have been included.

```
#include<stdio.h>
    void swap(char *str1, char *str2)
    {
   char *temp = str1;
   str1 = str2;
   str2 = temp;
   }

   int main()
  {
   char *str1 = "John Forbes";
   char *str2 = "Nash Jr";
   swap(str1, str2);
   printf("%s %s", str1, str2);
   getchar();
   return 0;
  }
```

(a) Nash Jr John Forbes
(b) John Forbes Nash Jr
(c) Forbes John Jr Nash
(d) John Jr Forbes Nash

5. Figure out the output of the below code. State relevant assumption if any during computation. Assume Necessary libraries have been included.

```
#include <stdio.h>
char fun();
int main(void)
{
      printf("%d %c\n", fun(), fun());
      return 0;
}
char fun()
{
      return 'G';
}
```

(a) 71 G
(b) G 71
(c) Error at compile time at line 4
(d) G G

6. Choose the correct option to fill ?1 and ?2 so that the program below prints an input string in reverse order. Assume that the input string is terminated by a newline character.

```
void reverse(void)
{
 int c;
 if (?1) reverse();
 ?2
}
int main()
{
 printf ("Enter Text ") ;
 printf ("\n") ;
 reverse();
 printf ("\n") ;
}
```

(a) ?1 is (getchar() != ’\n’)
    ?2 is getchar(c);
(b) ?1 is (c = getchar()) != ’\n’)
    ?2 is getchar(c);
(c) ?1 is (c != ’\n’)
    ?2 is putchar(c);
(d) ?1 is ((c = getchar()) != ’\n’)
    ?2 is putchar(c);

7. Consider the following C program that attempts to locate an element x in an array Y[] using binary search. The program is erroneous.

```
1. f(int Y[10], int x) {
2.   int i, j, k;
3.    i = 0; j = 9;
4.    do {
5.          k =  (i + j) /2;
6.          if( Y[k] < x)  i = k; else j = k;
7.       } while(Y[k] != x && i < j);
8.    if(Y[k] == x) printf ("x is in the array ") ;
9.    else printf (" x is not in the array ") ;
10. }
```

On which of the following contents of Y and x does the program fail?

(a) Y is [1 2 3 4 5 6 7 8 9 10] and x < 10
(b) Y is [1 3 5 7 9 11 13 15 17 19] and x < 1
(c) Y is [2 2 2 2 2 2 2 2 2 2] and x > 2
(d) Y is [2 4 6 8 10 12 14 16 18 20] and 2 < x < 20 and x is even

8. Consider the data given in above question Q.7, the correction needed in the program to make it work properly is

(a) Change line 6 to: if (Y[k] < x) i = k + 1; else j = k-1;
(b) Change line 6 to: if (Y[k] < x) i = k - 1; else j = k+1;
(c) Change line 6 to: if (Y[k] <= x) i = k; else j = k;
(d) Change line 7 to: } while ((Y[k] == x) && (i < j));

9. Consider the following C program

```
 int a, b, c = 0;
 void prtFun (void);
 int main ()
{
        static int a = 1; /* line 1 */
        prtFun();
        a += 1;
        prtFun();
        printf ( "\n %d %d " , a, b) ;
}

void prtFun (void)
{
        static int a = 2; /* line 2 */
        int b = 1;
        a += ++b;
        printf (" \n %d %d " , a, b);
}
```

What output will be generated by the given code segment?

(a) 3 1
    4 1
    4 2
(b) 4 2
    6 1

6 1
(c) 4 2
6 2
2 0
(d) 3 1
5 2
5 2

10. Consider the above question Q.9. What output will be generated by the given code \segment if: Line 1 is replaced by “auto int a = 1;” Line 2 is replaced by “register int a = 2;”

(a) 3 1
4 1
4 2
(b) 4 2
6 1
6 1
(c) 4 2
6 2
2 0
(d) 4 2
4 2
2 0

11. Consider the following C function in which size is the number of elements in the array E:

```
int MyX(int *E, unsigned int size)
{
    int Y = 0;
    int Z;
    int i, j, k;

    for(i = 0; i < size; i++)
         Y = Y + E[i];
         for(i = 0; i < size; i++)
              for(j = i; j < size; j++)  {
                   Z = 0;
                   for(k = i; k <= j; k++)
                          Z = Z + E[k];
                  if (Z > Y)
                      Y = Z;
              }
        return Y;
}
```

The value returned by the function MyX is the

(a) maximum possible sum of elements in any sub-array of array E.
(b) maximum element in any sub-array of array E.
(c) sum of the maximum elements in all possible sub-arrays of array E.
(d) D. the sum of all the elements in the array E.

12. Consider the following C function.

```
int fun1 (int n)
  {
   int i, j, k, p, q = 0;
   for (i = 1; i<n; ++i)
   {
            p = 0;
            for (j=n; j>1; j=j/2)
            ++p;
            for (k=1; k<p; k=k*2)
            ++q;
   }
   return q;
  }
```

Which one of the following most closely approximates the return value of the function fun1?

(a) n3
(b) n (logn)2
(c) nlogn
(d) D. nlog(logn)

13. Suppose you are provided with the following function declaration in the C programming language.

int partition (int a[], int n);

The function treats the first element of a[] as a pivot, and rearranges the array so that all elements less than or equal to the pivot is in the left part of the array, and all elements greater than the pivot is in the right part. In addition, it moves the pivot so that the pivot is the last element of the left part. The return value is the number of elements in the left part. The following partially given function in the C programming language is used to find the kth smallest element in an array a[ ] of size n using the partition function. We assume k ≤ n

```
int kth_smallest (int a[], int n, int k)
{
   int left_end = partition (a, n);
   if (left_end+1==k)
   {
       return a [left_end];
   }
   if (left_end+1 > k)
   {
      return kth_smallest (____________________);
   }
   else
   {
      return kth_smallest (____________________);
    }
}
```

The missing argument lists are respectively

(a) (a, left_end, k) and (a+left_end+1, n–left_end–1, k–left_end–1)
(b) (a, left_end, k) and (a, n–left_end–1, k–left_end–1)
(c) (a, left_end+1, N–left_end–1, K–left_end–1) and(a, left_end, k)

(d) D. (a, n–left_end–1, k–left_end–1) and (a, left_end, k)

14. Consider the following C program. The output of the program is ___________.

```
#include <stdio.h>

int f1(void);
int f2(void);
int f3(void);
int x = 10;

int main()
{
    int x = 1;
    x += f1() + f2() + f3() + f2();
    printf("%d", x);
    return 0;
}

int f1()
{
    int x = 25;
    x++;
    return x;
}

int f2( )
{
    static int x = 50;
    x++;
    return x;
}

int f3( )
{
    x *= 10;
    return x;
}
```

(a) 230
(b) 131
(c) 231
(d) 330

Problems for Programming Practice

(After the successful studies of Lecture 05 (Functions in C), the students are supposed to solve the following problems in C programming language.)

1. Two numbers are said to be co-prime, if the greatest common divisor of the numbers is one. For examples, 13 and 14 are co-prime but 14 and 21 are not. Write a C function *void CoPrime(int a, int b)* to test whether the pair of numbers *a* and *b* are co-prime.

In the main program, read five numbers and use this function to test how many of them are co-prime.

2. Write a C function say void GuessMe( ) that assumes a number between 1 and 100 inclusive. In your function, you can call *int rand( )* function, which is defined in C library to assume such a number. From your function prompt the user to guess the number and enter the number he has guessed. If the user guesses the number correctly, print out a congratulatory message and exit. If the user makes a wrong guess, the user is given an opportunity to guess again. The process may be repeated till user guesses correctly. Print the number of guesses a user makes to get the correct value.

3. A point in a plane can be represented by two coordinate values (x, y).
   a) Write C functions to do the following:
      i) Read a set of points (maximum 10).
      ii) Find the distance between a pair of points.
      iii) Whether any three given points constitute an isosceles triangle and if so find the area of the triangle.
      iv) Whether four points constitute a rectangle and if so find the area of the rectangle.
   b) Call these functions from the main program to find the number of isosceles triangles and rectangles that can be formed by the selected points and then find the one having maximum area in each case.

4. Write the following functions.
   (a) *int max (int *a)*: To find and return the maximum number in the array *a*.
   (b) *int min (int *a)*: To find and return the minimum number in the array *a*.
   (c) *int avg (int *a)*: To find and return the average of all the numbers in the array *a*.

   In the main function you do the following:
   i) Define an array say *marks* of type integer of size 100.
   ii) Read *n* ($0 \le n \le 100$) from the keyboard.
   iii) Read *n* numbers form the keyboard and store them into the array.
   iv) Call the functions *max(…)*, *min(…)* and avg(…) to calculate the maximum, minimum and average of the numbers in the array.
   v) Print the array marks and the results returned by the functions.

5. n-th ($n \ge 2$) Fibonacci number is defined as follow;
   $F_n = F_{n-1} + F_{n-2}$ with $F_1 = F_0 = 1$

   Write a recursive function *int Fibonacci(int n)* , which would return the n-th Fibonacci number when you call it from main for an input *n* .

6. Read any two positive numbers *a* and *b*. Write a recursive function *int gcd(int a, int b)*, which would return the greatest common divisor of *a* and *b*. You should read the numbers *a* and *b* in the main function and call *gcd(a,b)* in the main.

7. The n-th Harmonic number is defined as $H_n = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots$. Write a recursive function to calculate H_n, for any integer $n > 0$.

8. Write C functions to perform the following operations with two-dimensional arrays.
   i) Reading any two dimensional array elements.
   ii) Finding maximum and standard deviation of array elements.
   iii) Printing the transpose of a matrix

   Test these functions by calling them from the main program with 2d-arrays having maximum dimension 3 x 4.

9. Consider a three dimensional vector space over integer. Write C functions to perform following vector operations.
   i) Dot product of two vectors.
   ii) Cross product of two vectors.

   Read a set of vectors (maximum 10) and using the functions thus created, find how many of these vectors are perpendicular and parallel. Also find the maximum area of the parallelogram that can be formed by two vectors from the set of vectors read.

10. A complex number can be represented as a structure having two components corresponding to its real and imaginary values. Define a structure say "Complex" to store any complex number.
    Use you structure definition to write the following C functions:
    a) Complex AddComplex (Complex Z1, Complex Z2); // To add two complex numbers Z1 and Z2 and return the result.
    b) Complex SubtractComplex (Complex Z1, Complex Z2); // To subtract Z2 from Z1 and return the result.
    c) Complex MultiplyComplex (Complex Z1, Complex Z2); // To multiply two complex numbers Z1 and Z2 and return the result.
    d) Complex DivideComplex (Complex Z1, Complex Z2); // To dicvide Z1 by Z2 and return the result.
    e) float ModulusComplex (Complex Z); // To return the modulus of the complex number Z.
    f) float InverseComplex (Complex Z); // To return the inverste of the complex number Z.
    g) float ArgumentComplex (Complex Z); // To return the argument of the complex number Z.

    You should read complex numbers from the main and then call the above-mentioned functions and then print the results.

Use the appropriate functions to find the polar representation of a complex number.

Read a quadratic equation $Ax^2 + Bx + C = 0$ and the find its reots and store them as the complex numbers.

11. Determine what each of the following foomatic functions computes:

```
------------------------------------------------------------

    unsigned int foo1 ( unsigned int n )
    {
       unsigned int t = 0;

       while (n > 0) {
          if (n % 2 == 1) ++t;
          n = n / 2;
       }
       return t;
    }

---------------------------------------------------------------

    unsigned int foo2 ( unsigned int n )
    {
       unsigned int t = 0;

       while (n > 0) {
          if (n & 1) ++t;
          n >>= 1;
       }
       return t;
   }

 -----------------------------------------------------------------

    double foo3 ( double a , unsigned int n )
    {
       double s, t;

       s = 0;
       t = 1;
       while (n > 0) {
          s += t;
          t *= a;
          --n;
       }
       return s;
    }

---------------------------------------------------------

    double foo4 ( float A[] , int n )
    {
       float s, t;

       s = t = 0;
       for (i=0; i<n; ++i) {
          s += A[i];
          t += A[i] * A[i];
    }
    return (t/n)-(s/n)*(s/n);
```

12. Write functions to compute the following.
    a) The area of a circle whose diameter is supplied as an argument.
    b) The volume of a 3-dimensional sphere whose surface area is given as an argument

    For each of the above functions write a main program to read the input to the function and print the output of the function for that input.

13. Two rectangles are said to *overlap* if there exists a common point lying inside or on the boundary of both rectangles. Assume that all rectangles have edges parallel to the x and y axes.

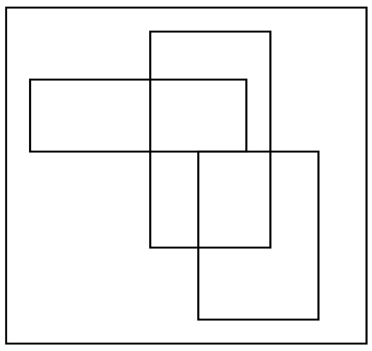
Fig 1 (a)

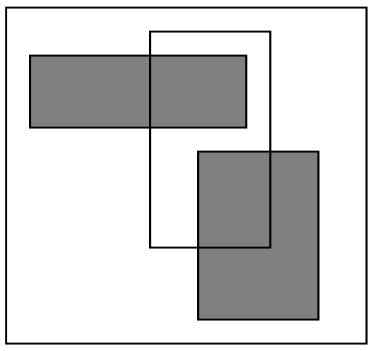
Fig 1 (b)

Fig 1(a) shows a case where all pairs of rectangles overlap. In Fig 1(b). the shaded rectangles do not overlap. Both figures contain 3 rectangles.

a) Write a C function ***overlap(x1, y1, x2, y2, a1, b1, a2, b2)*** that receives the diagonally opposite corner points of two rectangles as arguments such that *(x1, y1)* and *(x2, y2)* are the two diagonally opposite corner points of one rectangle, while *(a1, b1)* and *(a2, b2)* are the two diagonally opposite corner points of the other rectangle. The function returns 1 if the rectangles overlap, otherwise 0.

b) Write a ***main( )*** program that reads the value of an integer ***N*** and then reads in the diagonally opposite corner points of *N* rectangles. Assume *N* < 10. The coordinates are stored in four arrays, *X1[ ], Y1[ ], X2[ ], Y2[ ]*. The coordinates *(X1[k], Y1[k])* and *(X2[k], Y2[k])* represent the corner points of the kth rectangle. Your program must then use the function ***overlap(…)*** to determine whether all pairs of rectangles overlap. If not, then it must print the corner points of every distinct pair of rectangles which do not overlap.

--*--